AMENDMENTS TO THE SPECIFICATION

Please amend the specification as follows:

Please replace paragraph [003] with the following amended paragraph:

[003] Modern networks have facilitated the ability to broadly transfer digital information to and from various locations. One way of making data available for transport is by creating a static file such as an HTML file that can be downloaded from the server to a client computer system where it can be viewed. Static files can be cached in a variety of locations including at a server computer system, in a regional database, or even locally on the client computer system. When computer system static files are stored at a client computer system, retrieval and display by the client computer system is, for all practical purposes, instantaneous. Even when static files are stored directly on a server computer system connected to the client computer system (commonly known as output caching), retrieval of cached content can occur relatively quickly. Further, because of their static nature, static files can be cached for long periods of time, essentially indefinitely.

Please replace paragraph [009] with the following amended paragraph:

[009] Some programming frameworks support cache dependencies that cause a ~~cashed~~cached value to be dependent on either a time, a file or a key. A time cache dependency causes an entry associated with the specific dependency to be purged after the expiration of a specific period of time. For example, the Cache.Insert instruction can be implemented as follows to cause data to be dependent on time:

Cache.Insert(strCacheKey, dataset, nothing, DateTime.Now.AddMinutes(60), TimeSpan.Zero, CacheItemPriority.High),

More specifically, the DateTime.Now.AddMinutes(60) instruction causes an entry in the cache to be purged from the cache after 60 minutes. Using a time cache dependency that is set too short can result in the too frequent purging of the cache and the added cost of frequently fetching data from a database or the cost of calculating data to be stored in the cache. If the time value is set to long, there is the risk that the data stored in the cache will be invalid when it is delivered to a requesting client computer system.

Please replace paragraph [015] with the following amended paragraph:

[015] It may be that a cache entry is invalidated based on a customized cache dependency. A server computer system (e.g., including a Web page server) receives a notification that a monitored event has occurred. The notification can originate from a notify dependency changed method that was executed as a result of a creating a customized cached dependency and that monitors for the occurrence of an event associated with the customized cached dependency. In response to the notification, the server computer system invalidates (and removes) the cache entry. The server computer system calls a dependency dispose method to release any resources used by the cache entry.

Please replace paragraph [024] with the following amended paragraph:

[024] The extensible CacheDependency base class includes two additional methods that are publicly available such that developers of customized CacheDependency classes can invoke the purging capabilities of the extensible CacheDependency base class. The first method is a NotifyDependencyChanged method. This method is called by classes that are derived from the extensible CacheDependency base class to tell the extensible CacheDependency class that the dependent item has changed. The second new method is the DependencyDispose method. This method is a virtual method that provides a location cleanup specific to the derived class. More specifically, this method provides a location for instructions that normally would have been placed in the Dispose method of the ~~extenda~~extensible CacheDependency base class were it not for the customized CacheDependency class. As such, resources used by the customized CacheDependency class can be released for use by other modules having need of access to the resources.

Please replace paragraph [025] with the following amended paragraph:

[025] The NotifyDependencyChanged method may be called by a customized CacheDependency class that depends from the extensible CacheDependency base class. The customized CacheDependency class can thus cause a cache entry to be dependent on changes in a database, on data of a Web service, or for any other reason. In this way, the cache can recognize an invalid cache entry key and remove the key (and corresponding content) from the cache.

Please replace paragraph [026] with the following amended paragraph:

[026] In one illustrative example, a developer creates a custom class "SqlCacheDependency" that derives from the extensible CacheDependency base class to poll an SQL database to determine if changes have been made to the database. The SqlCacheDependency class is derived from the extensible CacheDependency base class that exists within a programming framework. Thus, the class SqlCacheDependency can call the NotifyDependencyChanged method such that a server computer system causes a dependent cache entry to be purged.

Please replace paragraph [028] with the following amended paragraph:

[028] Line 10 of the above code checks to see if an entry "Product" is already in cache. If the entry is not in cache, then the functions in the "if" statement are performed. Line 20 establishes a connection to a database where content is to be ~~retireived~~retrieved. Line 30 retrieves data from the database using a method "Product_GetProducts" and stores the data in a variable "ds." At line 40 the ~~varioable~~variable "~~ds~~dep" is assigned as a dependency on the "Products" table in the "NorthWind" database. This dependency is a customized SqlCache~~Depende~~yDependency that derives from the extensible CacheDependency base class. Line 50 then causes the data in the variable "ds" to be inserted into a cache entry referenced by the key "Product" such that the cache entry is dependent on "dep".

Please replace paragraph [029] with the following amended paragraph:

[029] In this example, SqlCacheDependency is a customized cache dependency derived from the extensible CacheDependency base class. When a class deriving from the extensible CacheDependency base class is used as a dependency, the server computer system processes the

class as if it were an instance of the extensible CacheDependency class. Accordingly, the server computer system is able to respond to similar behaviors exhibited by the extensible CacheDependency base class, while internally the derived class may implement its own unique functionality, such as being notified when a database query changes. The customized CacheDependency class, in this case SqlCacheDependency, overrides any methods of the extensible CacheDependency base class necessary to achieve the desired functionality. In this example, when SqlCacheDependency is notified of the database query change, it will call the base.Notify.DependencyChanged method causing the server computer system to remove the cache entry referenced by the key "Product".

Please replace paragraph [032] with the following amended paragraph:

[032] Referring now to Figure 1, a suitable environment where aspects of the present invention may be practiced is shown. Figure 1 depicts a server 100 and client 102 connected to a network 104 by corresponding links 121 and 122 respectively. The server 100 and client 102 can communicate over network 104 using a variety of protocols including HTTP. The server 100 can be an ASP.NET server, such that it can deliver dynamic content in the form of dynamic Web pages to the client 102. The server 100 includes various classes and modules including an extensible CacheDependency class 106 and a customized CacheDependency class 114 . The server 100 also includes a cache 110 that may be located in the physical memory of the server 100 or in any other convenient location.

Please replace paragraph [034] with the following amended paragraph:

[034] Illustratively, the client 102 requests certain content from the server 100. For example, client 102 may access a URL that requests a Web page from the server 100. The server 100 checks the cache 110 to determine if the requested content (e.g., data in a Web page) is in the cache 110. If the content is not in the cache 110, the database interface module 108 can request the content from the database 112. When appropriate, content, such as, for example content 132, is retrieved and returned to a database interface module 108.

Please replace paragraph [037] with the following amended paragraph:

[037] The customized CacheDependency class 114 may cause the cache 110 to be purged when there is a change in the database 112. In this example, while fetching data from the database 112 may be resource intensive, polling the database 112 for changes, on the other hand, uses less resources. The database 112 may comprise database tables that store information in the database 112 such as the database table 116. Another database table 117 may further comprise a flag entry 118 that indicates if changes have been made to information in the database table 116. The database table 117 may contain flags for other tables in the database that have custom dependencies. In one embodiment, the flag entry 118 may be incremental such that each change in the database table 116 causes the flag entry 118 to incrementally increase its value. The customized CacheDependency class 114 may be designed to remove cache entries when an increase in the flag entry 118 is detected.

Please replace paragraph [039] with the following amended paragraph:

[039] The method of Figure 2 includes deriving a customized CacheDependency class from the extensible CacheDependency base class (act 204). Act 204 can include a server computer system deriving a customized CacheDependency class from the extensible CacheDependency base class. For example, server 100 can derive customized ~~cache dependency~~CacheDependency class 114 from extensible CacheDependency class 106. Server 100 may derive customized ~~cache dependency~~CacheDependency class 114 as a result of executing instructions developed by a web page developer, software developer or as part of some program module, to create a customized CacheDependency. Server 100 may also derive customized ~~cache dependency~~CacheDependency class 114 from a number of sources, including but not limited to: an external sensor connected to a computer, such as a thermometer, that triggers an event whenever conditions change; a sports scoreboard program that triggers a change whenever the score in a game changes; a network management program that triggers an event whenever the network configuration changes (such as a computer being added or removed from the network); and a directory program that triggers an event whenever the directory changes (such as a person being added or removed).

Please replace paragraph [040] with the following amended paragraph:

[040] The method of Figure 2 includes accessing content (act 206). Act 206 can include a server computer system accessing content that is to be included in a Web page for delivery to a client computer system. For example, server 100 can access content that is to be included in a Web page for delivery to client 102. Accessing content can include creating or manipulating the content. Content can be accessed from a variety of different locations, such as, for example, from databases and Web services. For example, server 100 can access data stored at database 112. Database interface module 108 or other suitable modules can appropriately format the data for inclusion in a Web page.

Please replace paragraph [041] with the following amended paragraph:

[041] The method of Figure 2 includes an act of creating a cache entry that is associated with the customized CacheDependency class (act 208). Act 208 can include a server computer system creating a cached entry that is associated with the customized CacheDependency. For example, server 100 can create a cache entry that is associated with customized ~~cache dependency~~CacheDependency class 114. A cache entry is created in one embodiment by a variable definition statement made by the server 100 (e.g., similar to line 50 of the above described instructions).

Please replace paragraph [042] with the following amended paragraph:

[042] The method of Figure 2 includes an act of inserting the cache entry into cache (act 210). Act 210 can include a server computer system inserting the cached entry into a cache location at the server computer system. For example, server 100 can insert a cache entry into cache 100. A cached entry can be inserted into cache such that the validity of the cached entry (and thus also the cached content) is dependent on the customized dependency. For example, server 100 can insert a cache entry into cache 110 that is dependent on a dependency resulting from customized ~~cache dependency~~CacheDependency class 114.

Please replace paragraph [043] with the following amended paragraph:

[043] Referring now to Figure 3, a method for invalidating a cache entry using a customized CacheDependency is shown. The method in Figure 3 includes an act of monitoring custom dependency conditions set by an instance of a customized cache dependency (act 302). Act 302 can include a server computer system monitoring custom dependency conditions set by an instance of a customized CacheDependency class implemented at the ~~sever~~server computer system. For example, server 100 can monitor custom dependency conditions set by an instance of a customized CacheDependency class 114. A customized dependency condition may be the occurrence of an event, such as, for example, a change in the content of a database or in the content provided by a Web service. For example, server 100 can monitor flag entry 118 for changes in the content of database table 116. When customized CacheDependency class 114 is associated with an aggregate dependency, a plurality of custom dependency conditions can be monitored.

Please replace paragraph [044] with the following amended paragraph:

[044] The method of Figure 3 includes an act of determining if the custom dependency condition has been satisfied (act 304). Act 304 can include a server computer system determining if a custom dependency condition associated with a customized cached dependency has been satisfied. For example, server 100 can determine if a custom dependency condition associated with customized CacheDependency class 114 has been satisfied. When customized CacheDependency class 114 is associated with an aggregate dependency, server 100 can determine if each custom dependency condition in a plurality of custom dependency conditions has been satisfied.

Please replace paragraph [047] with the following amended paragraph:

[047] The method of Figure 3 includes an act of indicating that custom dependency conditions have been satisfied (act 306). Act 306 can include a NotifyDependencyChanged method indicating to a server computer system that the custom dependency conditions have been satisfied. For example, a NotifyDependencyChanged method for customized ~~CachDependency~~CacheDependency class 114 can notify server 100 when content ~~it~~in database

table 116 changes. Notification that a custom dependency condition has been satisfied can indicate to server 100 that a cache entry is to be purged.

Please replace paragraph [048] with the following amended paragraph:

[048] The method of Figure 3 includes purging the cache (act 308). Act 308 can include a server computer system purging a cache entry at the server computer system. For example, server 100 can purge a cache entry from cache 110. A cached entry can be purged in response to receiving a notification that custom dependency conditions associated with a customized cache dependency have been satisfied. For example, server 100 may purge a cached entry dependent on data table 116 in response to receiving a notification that flag entry 118 was incremented. When a cache entry is purged, computer system 100 can release any resources (e.g., system memory) that were being consumed to maintain the cache entry. For example, server computer system 100 can invoke a DependencyDispose method associated with customized ~~CachDependency~~CacheDependency class 114 to release resources for a cache entry that stored content from database table 116.